iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** V **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.82399

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

A Secure Web-Based Investment Club Management System with Integrated Accounting and Role-Based Access Control

Ayodeji Bolanle Balogun[1], Babajide Adelaja Ojo[2]
[12]Dept. of Electrical Electronic Engineering, Federal Polytechnic Ilaro, Ogun State, Nigeria

Abstract: ***Cooperative savings and investment clubs play a crucial role in domestic finance in sub-Saharan Africa, but most rely on handwritten records that are error-prone, vulnerable to fraud, and easily lost. This paper describes the design and development of an online Investment Club Management System (ICMS) that digitises the entire operating cycle of a cooperative savings and loan society. The system integrates type-safe client–server communication, role-based access control, and a structured Know Your Customer onboarding process. A four-stage loan workflow is implemented with capacity-aware guarantor allocation, alongside automated amortisation and double-entry accounting within a general ledger framework. The proposed design improves data integrity, operational transparency, and financial reporting accuracy. Experimental evaluation demonstrates reliable performance and consistency in transaction processing. The system enhances transparency and efficiency and provides an architectural template for developing secure financial management systems for community organisations.***
Keywords: cooperative society management, role-based access control, Know Your Customer, loan management, double-entry accounting, type-safe remote procedure calls

I. INTRODUCTION

Millions of households across Nigeria and the rest of sub-Saharan Africa are outside the formal financial sector and rely on cooperative savings and lending groups. In Nigeria they are commonly termed as ajo, esusu, or adashe (depending on location) and collect monthly subscriptions, provide interest-based loans and pay dividends from any surplus (Okonkwo et al., 2023; Adeola et al., 2022). These practices result in a large transactional data set that is difficult to manage. Ledger entries are open to data entry errors and fraud; loan approval processes lack audit trails; guarantor activities are handled informally and financial reports (when generated) are not prepared in accordance with accounting standards (Maliwa & Simatela, 2025; Nwoye et al., 2025).

Technology has started to resolve these issues. Recent research shows the technical feasibility of web-based cooperative management systems (Hakiki et al., 2025; Nwoye et al., 2025), but published systems tend to have limited functionality or lack formal integration with accounting systems, identity verification or guarantor management. This paper addresses these challenges with the following contributions:

1) A complete, end-to-end system architecture integrating KYC onboarding, multi-tier Role-based Access Control (RBAC), a four-stage loan pipeline with guarantor capacity enforcement, double-entry accounting, and financial reporting in a single cohesive application.
2) A structured guarantor governance mechanism that prevents self-guarantorship and limits each member to guaranteeing at most two active loans simultaneously, a rule not addressed in any prior published system.
3) A demonstration that a type-safe, full-stack TypeScript architecture (React + tRPC + Drizzle ORM) is a practical and reliable framework for cooperative financial software.
4) A qualitative evaluation against thirteen functional requirements, supported by expert review.

II. RELATED WORK

Hakiki et al. (2025) developed a web system for savings and loan cooperatives with a payment gateway for loans and deposits. They relieved the management problems but did not formalise the double-entry accounting and their role-based access control was binary. Nwoye et al. (2025) implemented the Mgbakwu Cooperative Management System with React and Flask which included loan amortisation and RBAC but not KYC checks, management of guarantors or dividend payment. Maliwa and Simatela (2025) developed a loan application and approval system for a cooperative in Zambia, but did not address member contributions or reporting.

In terms of security, Wen et al. (2009) suggested a three-layered RBAC model for large financial web systems, showing that layered access control minimises the risk of privilege escalation. Kawamura (2022) recognised the trade-off between compliance and convenience in online KYC processes, while Komandla (2018) suggested asynchronous document verification to minimise user drop-offs, which is directly incorporated into the Investment Club Management System (ICMS). While Levente Kovács (2018) highlights the importance of structured financial intermediation and institutional control within banking systems, the study does not explicitly address loan amortisation mechanisms, particularly in terms of their computational implementation and integration within modern cooperative financial management systems.

Table 1 summarises the feature coverage of the most closely related systems alongside the ICMS, confirming that no prior published system addresses the full combination of features that the ICMS provides.

Table 1: Feature Comparison of Related Cooperative Management Systems

Feature	Hakiki et al. (2025)	Nwoye et al. (2025)	Maliwa & Simatela (2025)	ICMS
Member registration	Yes	Yes	Yes	Yes
KYC document verification	No	No	No	Yes
Multi-role RBAC (3+ roles)	No	No	No	Yes
Loan application workflow	Yes	Yes	Yes	Yes
Guarantor capacity enforcement	No	No	No	Yes
Loan amortisation scheduling	No	Yes	No	Yes
Contribution tracking	Yes	Yes	No	Yes
Dividend distribution	No	No	No	Yes
Double-entry general ledger	No	No	No	Yes
Financial statements	No	Partial	No	Yes
Immutable audit log	No	No	No	Yes

III. SYSTEM DESIGN AND IMPLEMENTATION

A. Requirements

A review of the operations of the cooperative society of Nigeria and the identified gaps in the related work led to the derivation of thirteen functional requirements. These are: three tier role based access control (admin, member, non-member); a Know Your Customer onboarding gate prior to financial access; administrative review of KYC submissions; loan application with two nominated guarantors; prohibition of self guarantorship; maximum of two active guarantorships per member; automatic generation of a loan amortisation schedule upon disbursement; journal recording by doubling the entries; creation of financial statements including Balance Sheet, Income Statement and Cash flow Statement; an immutable audit log; in-application and email notification; administrator-readable interest rates and loan fees; and member-initiated resubmission of rejected KYC submissions.

Moreover, the system meets the major non-functional requirements, such as security through enforced access control and auditability; data integrity via the use of double-entry accounting and immutable logs; scalability to meet increasing cooperative membership; availability through system access reliability; and maintainability through modular system design.

B. Architecture

The ICMS follows a three-tier client-server architecture. The presentation tier is a React 19 single-page application styled with Tailwind CSS 4 and the shadcn/ui component library. The application tier is an Express.js server exposing a tRPC API, which enforces end-to-end type safety by deriving client-side type definitions directly from server-side procedure definitions, eliminating the class of integration bugs that arise when a REST contract drifts from its implementation. The data tier is a MySQL database accessed through the Drizzle ORM. Figure 1 illustrates the architecture.

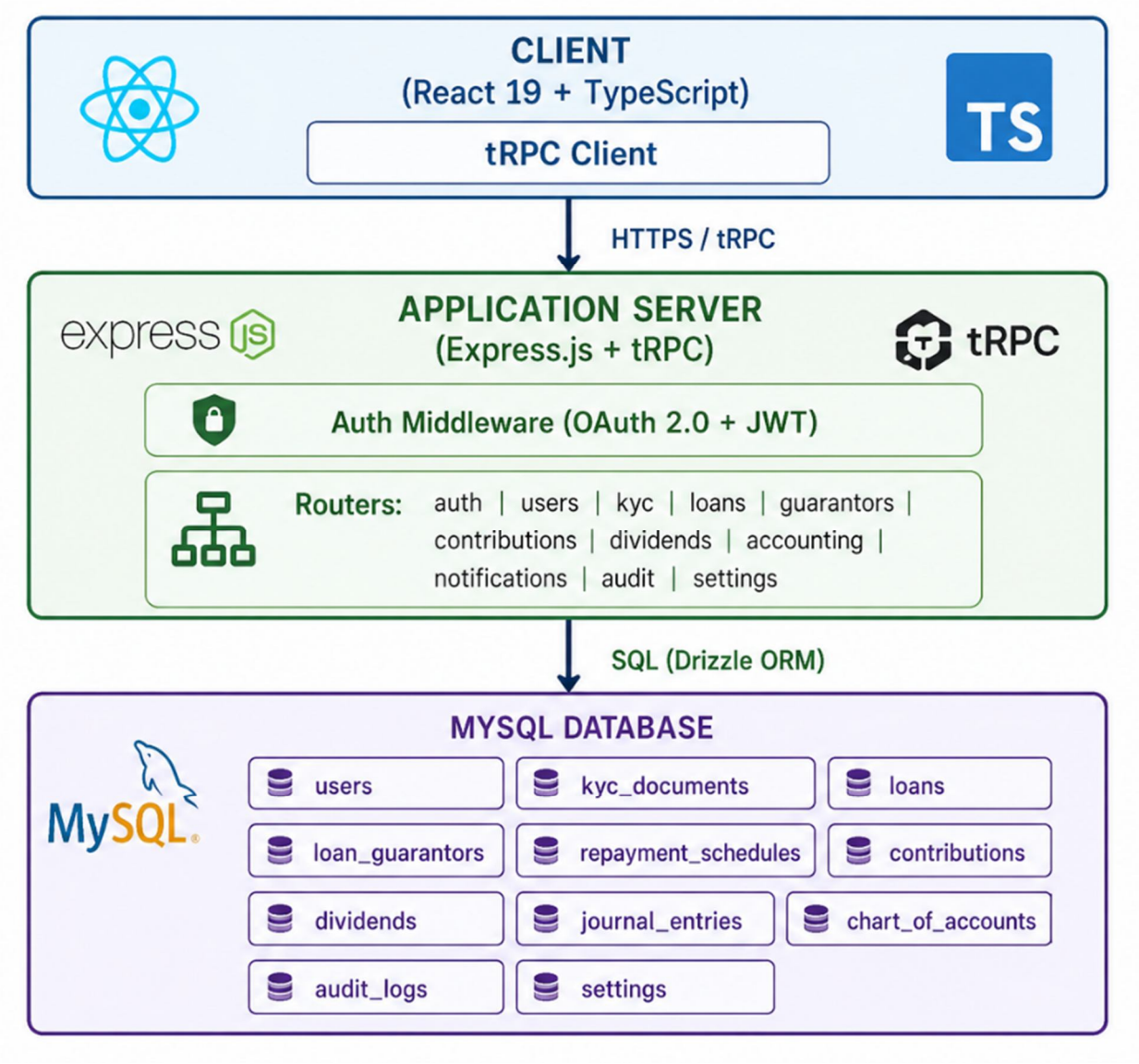


Figure 1. High-level architecture of the Investment Club Management System.

C. Technology Stack

The system is structured in such a way that it guarantees type safety, scalability, and data integrity at all levels. React 19 with TypeScript is used at the frontend, allowing a component-based architecture with type checking at runtime, minimizing the number of runtime errors and enhancing maintainability. Tailwind CSS and shadcn/ui help to design interfaces efficiently and consistently.

tRPC is used to implement client-server communication, with type safety across the entire system - end to end - without separate API schemas, eliminating the problem of data contract inconsistencies found in REST or GraphQL systems. Express.js is the preferred backend to be developed due to its lightweight design and flexibility.

MySQL handles data persistence, offering ACID-compliant transactions that can be used in financial records and in double entry accounting. Authentication is provided in OAuth 2.0 with JSON Web Tokens to secure, stateless session control. An S3-compatible object store can be used to store documents with scalability, and Vitest can be used to effectively test in the TypeScript environment.

This stack saves on redundancy, enhances reliability and offers uniformity in the processing of financial data. The main technologies employed are summarised in Table 2.

Table 2: Technology Stack

Layer	Technology	Purpose
Frontend	React 19, TypeScript 5, Tailwind CSS 4, shadcn/ui	Component UI and styling
API	tRPC 11	Type-safe remote procedure calls
Backend	Express.js 4	HTTP server
ORM	Drizzle ORM	Type-safe database access
Database	MySQL 8	Relational data storage
Auth	OAuth 2.0 + JWT	Session management
File storage	S3-compatible object store	KYC and loan documents
Testing	Vitest 2	Unit and integration tests

D. Role-Based Access Control

RBAC is implemented in two layers. tRPC middleware procedures (publicProcedure, protectedProcedure, adminProcedure) on the server validate the JWT session and role of the user before any business logic is executed. The adminProcedure checks roles (ctx.user.role = 'admin') and throws a FORBIDDEN error when an attempt is made to access unauthorised data. On the client side, the useAuth hook exposes information about roles to conditional rendering, and uses route guards to prevent unauthorised navigation.

Security wise, the design is based on some of the key recommendations provided by OWASP, especially in implementing access control on the server as the primary trust boundary, with frontend checks treated only as advisory.

In a simple threat model, the following risks have been identified:

- Privilege escalation: alleviated by server-side role verification on all the procedures protected and through strong separation of public and administrative endpoints.
- Broken access control: blocked by middleware enforcement that is centrally implemented and not spread out.
- Token misuse or session hijacking: reduced through secure JWT handling and controlled session validation.
- Direct URL access attacks: prevented through backend enforcement, so that route-level restrictions are not possible to evade.

In general, the system has a deny-by-default policy, in which the access can be provided only after explicit role verification. This is a layered strategy that minimises the attack surface and is consistent with the known principles of secure design of web-based financial systems.

E. KYC Onboarding Workflow

Upon the first login, the new user is redirected to an onboarding page where they provide personal information and an ID document. It is stored in an Amazon S3-compatible object store with only a reference key kept in the KYC documents table with a pending status. Until the user is given administrative approval, user accounts are inactive, after which an in-app notification is dispatched. In case of a rejection, the user can get the feedback of the reviewer and can resubmit the document using the profile interface as per the asynchronous review process described by Komandla (2018).

The system complies with data protection principles in accordance with the Nigeria Data Protection Regulation (NDPR), by ensuring data minimisation, secure storage and controlled access to personally identifiable information. Sensitive documents are not stored in the application database directly, and it reduces the exposure risk, in addition, access to KYC records is controlled by roles. Also, audit logging aids accountability and traceability in the process of handling user data as required by regulations on privacy and compliance.

F. Loan Application Pipeline

The loan application form is in four stages. Stage 1 is for loan amount and term. Stage 2 is for select guarantor: the applicant selects two active members (other than the applicant) from a list, and excludes a member who is guarantor of two active loans. Stage 3 is for the bank account of the borrower for loan disbursement. Stage 4 is for document uploads: a receipt for payment of the loan application fee (configured by the administration) and the applicant's most recent payslip.

The guarantor rules are implemented on both the front and back ends. On the back end, the applyForLoan procedure makes sure the borrower's id is not in the guarantor list, and calls getActiveGuarantorshipCount for each nominee, aborting if either count is two. On the front end, the eligibleGuarantors query returns the count for the borrower for each member, so that disabled members cannot be chosen in the drop-downs.

After the guarantors approve, admin approves the loan. When the loan is disbursed, an amortisation schedule is calculated (using standard reducing-balance method):

$$M = P \times r \times \frac{(1+r)^n}{((1+r)^n - 1)}, \qquad (1)$$

where P is the principal, r is the monthly interest rate, and n is the tenure in months. Each instalment is decomposed into interest and principal components and stored in the repayment_schedules table.

G. Double-Entry Accounting Module

Every financial event generates a corresponding journal entry enforced by the createJournalEntry helper, which rejects any entry whose debits do not equal its credits. Table 3 lists the principal transaction types and their accounting treatment.

Table 3: Accounting Treatment of Principal Transaction Types

Transaction	Debit	Credit
Member contribution received	Cash and Bank	Members' Contributions
Loan disbursed	Loans Receivable	Cash and Bank
Repayment received (principal)	Cash and Bank	Loans Receivable
Repayment received (interest)	Cash and Bank	Interest Income
Dividend distributed	Retained Earnings	Dividends Payable
Dividend paid out	Dividends Payable	Cash and Bank

The accounting model is congruent with the established principles of the balance and traceability of all financial transactions in the established principles of accounting that are defined in international financial reporting standards (IFRS) and generally accepted accounting principles (GAAP). Financial statements, the Balance Sheet, Income Statement and Cash Flow Statement are prepared through consolidating in the journal entries by account classification and a given reporting period.

To ensure correctness and reliability, the system provides the following internal validation rules: transaction balancing, account type consistency and chronological integrity of entries. Moreover, the audit log is immutable and thus allows tracking of all financial transactions in line with the verification and compliance needs.

Report generated can be exported into CSV and PDF format to enable external analysis and regulatory reporting.

IV. QUALITATIVE EVALUATION

A. Approach

The assessment has two parts: a traceability study of the requirements and an expert review. The requirements traceability analysis links each of the thirteen requirements to the system component that implements it, and checks implementation using the automated test suite (42 Vitest tests, all passing, zero TypeScript errors). The expert review asked three domain experts: a 12-year experienced treasurer of a cooperative society, a software engineer with expertise in financial systems, and an information systems academic. Each expert was given a demonstration of all major system workflows and rated the system on functional completeness, ease of use, data integrity, security and scalability.

B. Requirements Traceability

All thirteen functional requirements are satisfied. Table 4 presents a condensed traceability matrix.

Table 4: Requirements Traceability Matrix (Condensed)

Requirement	Implementing Component	Test Coverage
Three-tier RBAC	adminProcedure middleware; users.role column	6 RBAC tests
KYC onboarding gate	Onboarding page; is_active flag	6 KYC gate tests
Admin KYC review	kyc.review procedure; KYC Review page	Functional walkthrough
Loan application with guarantors	loans.applyForLoan; LoanApply form	4 loan validation tests
Self-guarantorship prohibition	Borrower-ID check in applyForLoan	1 guarantor rule test
Guarantor capacity limit (max 2)	getActiveGuarantorshipCount helper	1 guarantor rule test
Amortisation schedule on disbursement	loans.disburse; amortisation generator	Functional walkthrough
Double-entry journal entries	createJournalEntry helper	Functional walkthrough
Financial statements	accounting.balanceSheet/incomeStatement/cashFlow	3 accounting tests
Immutable audit log	writeAudit helper	Functional walkthrough
In-app and email notifications	createNotification; email alert procedures	2 notification tests
Admin-configurable settings	settings.update; Settings page	2 settings tests
KYC document re-submission	kyc.resubmit; Profile page	2 re-submission tests

C. Expert Review Findings

The treasurer of the cooperative society reported that the loan process in the application is the same as that of her society. In particular, she said, the guarantor capacity enforcement is important because disputes over the capacity of guarantors are often a matter of dispute in her society, and this automation will end this problem. She further asked for an amortisation schedule, which can be printed via the PDF support.

The software engineer stated that the application-level double-entry constraint is helpful but noted that it is not sufficient to prevent the ledger from becoming inconsistent without a corresponding database-level CHECK constraint, because any direct access to the database or unanticipated exception during a multi-step transaction could result in inconsistencies. The database-level constraint should be added in a future release.

The information systems researcher concurred that the system is suitable for small- to medium-sized cooperative societies and that the key architectural requirement to scale the system for a platform is multi-tenancy. She recommended a quantitative user-study with cooperative members to complement the qualitative feedback.

V. DISCUSSION

The fact that strict type safety is necessary throughout the application stack is what motivated the decision to use tRPC in financial systems where data inconsistencies can be the cause of material errors. The experience of ICMS development demonstrates that client interfaces that are based on server procedures eliminate the integration faults that are often seen in systems based on REST. Non-TypeScript clients, such as native mobile applications, would need more layers of abstraction or other API gateways, however.

Scalability wise, the present architecture is fine when dealing with moderate workloads but when dealing with large scale cooperative networks, it might need to be improved. To handle the large volume transactions and a number of concurrent users, horizontal scaling of the Express.js backend, database replication in MySQL, and addition of caching layers or message queues would be required. Multi-tenancy is also not implemented, thus restricting its use to federated cooperative models.

Security threats are a very sensitive issue. Where role-based access control and audit logging exist, there is a need to take further steps to be fully in line with the best practices as promoted by OWASP. They comprise resisting injection attacks, better token lifecycle management, encryption of sensitive information at rest and in transit, and overall monitoring of abnormal activities. The need to ensure the close follow-up of the data protection rules and rules of safe storage also emerges as a result of the processing of KYC data. The Nigerian situation is more of a challenge in terms of deployment. The sporadic internet connectivity especially in the rural areas affects the accessibility of systems and user experiences. This limits the application of full online architectures and suggests the need to have offline capable or synchronisation based architectures.

A lack of power stability, infrastructure constraints and differences in digital literacy levels of cooperative members are also barriers to adoption. The opportunity to connect with local payment providers, such as Paystack or Flutterwave, would make the service easier to use, yet it also implies a number of additional considerations about the reliability of API usage, reconciling transaction, and compliance with regulations.

The main constraints of this work thus go beyond the scope of evaluation. Though the system is functionally complete, the practical applicability of the system to the real world will depend upon the capability of overcoming the scalability limits, enhancing the security posture, and adjusting the deployment strategies to the local infrastructural conditions.

VI. CONCLUSION

The design and implementation of a web-based Investment Club Management System that supports the lifecycle of operation of cooperative savings and lending activities have been presented in this paper. The system combines KYC onboarding, role-based access control, structured loan pipeline with constraint on guarantor capacity, automatic amortisation scheduling, and double-entry accounting all in a single and type-safe full-stack architecture. A qualitative assessment shows that the specified functional requirements are addressed and that the system can be used by the small and medium-sized cooperative societies.

Its key contributions are the formalisation of guarantor governance and implementation of double-entry accounting within the workflow, which fill in gaps in current cooperative management systems. Future development will involve adding functionality to interface with local payment systems, support enforcement of accounting constraints on a database level, and a quantitative usability test with end users.

This work is relevant to the digital transformation of informal and semi-formal financial systems in developing economies and is applicable to cooperative societies in search of a better way to achieve transparency, accountability, and operational efficiency.

REFERENCES

[1] Adeola, O., Adeleye, I., Muhammed, G., Olajubu, B. J., Oji, C., & Ibelegbu, O. (2022). Savings Groups in Nigeria. Transforming Africa, 193–216. https://doi.org/10.1108/978-1-80262-053-520221015

[2] Hakiki, R., Delianti, V. I., Marta, R., Cabanillas-García, J. L., Slavov, V. D., & Jalil, S. ‘Afiat. (2025). Smart Financial Management for Cooperatives: A Web and Payment Gateway Integration Approach. Journal of Hypermedia & Technology-Enhanced Learning, 3(1), 17–37. https://doi.org/10.58536/j-hytel.161

[3] Kawamura, L. (2022). How can Financial Service Providers improve the KYC onboarding experience?: challenges and technological solutions. Www.theseus.fi. https://www.theseus.fi/handle/10024/752180

[4] Komandla, V. (2018). Transforming Customer Onboarding: Efficient Digital Account Opening and KYC Compliance Strategies. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.4983076

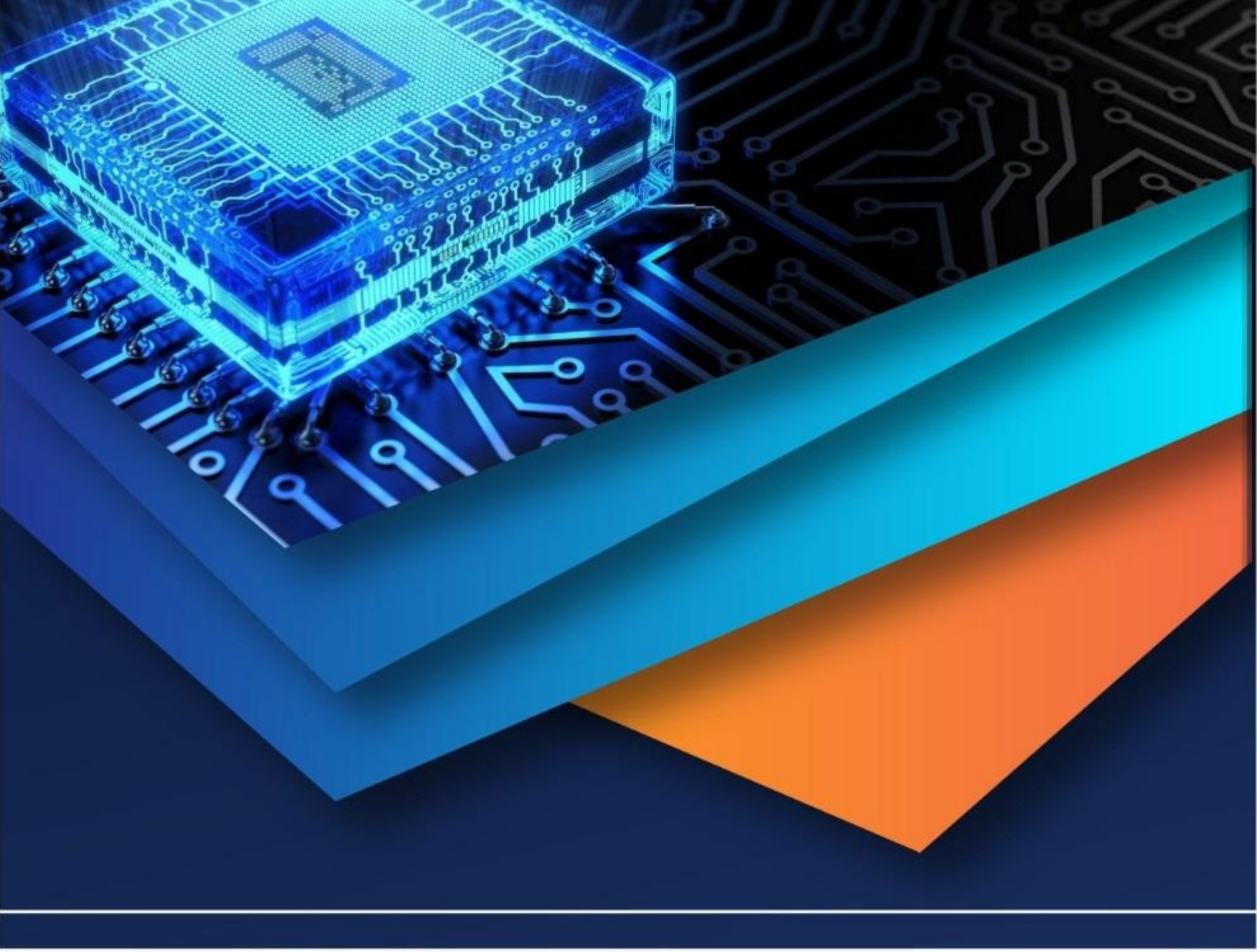
[5] Kovács, L. (2018). Loan Amortisation Algorithm Types, Amortisation Characteristics and Their Financial Implications. https://bankszovetseg.hu/Public/gep/2018/ENG/279-298%20Kovacs%20Levente.pdf

[6] Maliwa, M., & Simatela, M. (2025). Design and Development of a Co-Operative Society Information Management System. Scientific Journal of Engineering and Technology., 2(1), 11–22. https://doi.org/10.69739/sjet.v2i1.267

[7] Nwoye , J. N., Mmaduakonam, N. E., & Ezeanya, I. H. (2025). Design and Implementation of a Web-Based Management System for Mgbakwu Cooperative Society. Journal of Computer Science Review and Engineering, 9(3), 10–18. https://doi.org/10.5281/zenodo.17972078

[8] Okonkwo, C. J., Otaokpukpu, N. J., & Okafo, O. (2022). Analysis of Platform Cooperative and FinTech Value Proposition: The Microfinance Cooperative Digitalization take-away. 6th ICOPRON CONFERENCE, OGUN 2022. https://www.researchgate.net/publication/372290023

[9] Wen, Z., Zhou, B., & Wu, D. (2009). Three-Layers Role-Based Access Control Framework in Large Financial Web Systems. 2009 International Conference on Computational Intelligence and Software Engineering, 1–4. https://doi.org/10.1109/cise.2009.5362682



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY